Eunice Worifah

student id: 260560384

McGILL DESAUTELS

**INSY 661: Individual Project**

**AN ANALYSIS OF CORPORATE TRAVEL IN BRAZIL**

**2019 and 2020**

**PART 1**

***OVERVIEW***

A leading expense management company in Latin America has contacted me to help analyze corporate travel data in Brazil in 2019 and 2020 using SQL to facilitate the process. I am committed to obtaining insights which the company can use to identify patterns in corporate travel activity pertaining to all the stakeholders involved (the employees, employers, travel agencies, and hotels). This data can then be shared with the different stakeholders to drive their business decisions.

To undergo this process, I collected data that outlined 148,397 flight trips and 22,255 hotel reservations from 1335 employees who work at the 5 biggest corporations in Brazil. Using this data, I will produce insights that different stakeholders can use to improve decision-making in relation to their business goals.

**DATA PREPARATION**

The dataset I obtained was very large, and included information on travel in 2017, 2018, and even predictive data for 2021, 2022, and 2023. My goal was to analyze historical data and generate my own insights, so I decided to DROP all data except those relating to 2019 and 2020. I was selective and kept relevant data only. An example of the SQL query I used is:

DELETE FROM hotel

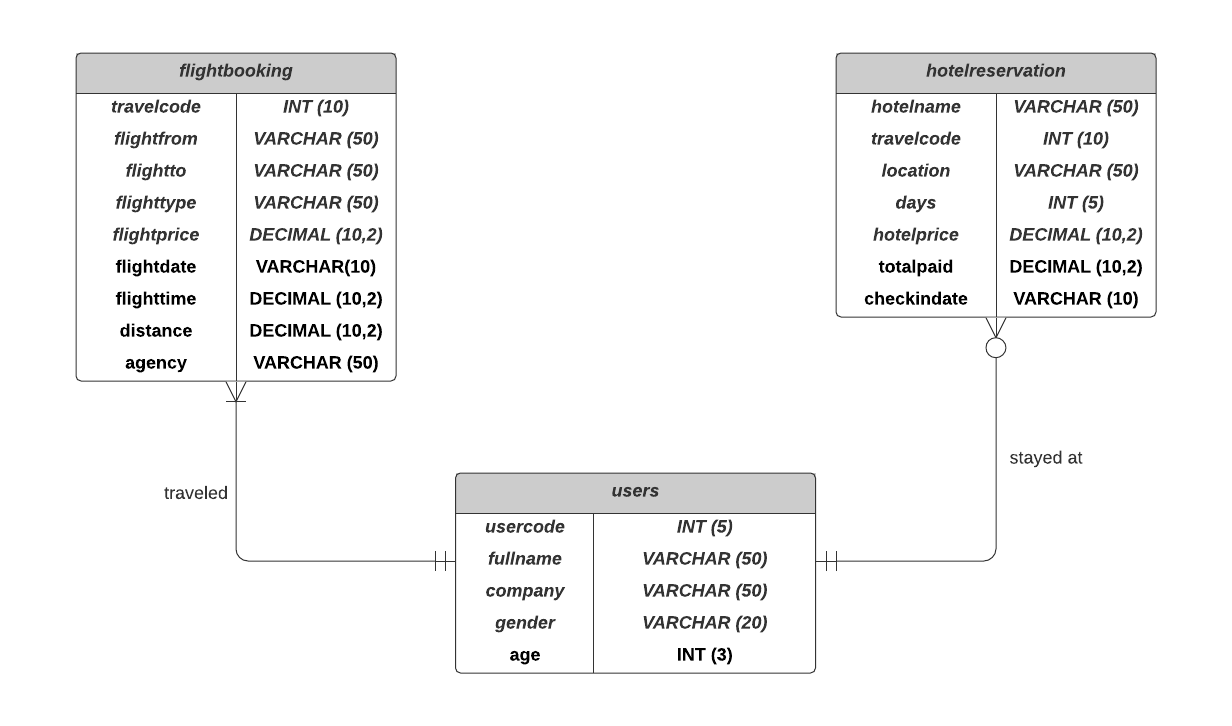
WHERE checkindate REGEXP '/2022'

To get the data ready for analysis, I had to change the column names and edited the format of certain data types. For example, the way the DATE was entered (01/01/2019), phpMyAdmin was unable to read it as a DATE. So, I changed the variable type to VARCHAR(10). It still contained the same information, but I was able to efficiently manipulate SQL to fit my needs.

I chose this dataset because as an avid traveller myself, and someone who is hoping to get a job which lets me travel frequently, it is interesting to gain an understanding of how much corporations spend on corporate travel every year. It was a very interesting project to work on!

**PART 2**

**ERD**



**ASSUMPTIONS**

* + - 1. Each user can have many flight bookings (could have completed multiple business trips) and must have at least one flight booking on file to be in the database.
      2. Each flight booking must be related to only one user (mandatory)
      3. Each user can have many hotel reservations (could have completed multiple overnight business trips), but does not have to have a hotel reservation (e.g. went on a one-day business trip).
      4. Each hotel reservation must be related to only one user (mandatory)

**DATA DICTIONARY**

**Description of Entities**

|  |  |  |  |
| --- | --- | --- | --- |
| **Entity Name** | **Description** | **Aliases** | **Occurrence** |
| users | Contains information needed to identify individuals/employees who went on business trips | employee | One user may make multiple flight bookings (mandatory) and may have multiple hotel reservations (optional) |
| flightbooking | Contains information needed to describe the flight that was taken by the user | flight booking | One flight booking may have been made by one user only (mandatory) |
| hotelreservation | Contains information needed to describe the hotel stay taken by the user during the business trip | hotel reservation | On hotel reservation may have been made by one user only (mandatory) |

**Description of Attributes**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Entity Name** | **Attributes** | **Description** | **Data Type** | **Nulls** | **Multi-valued** | **Derived** | **Default** |
| **users** | **usercode** | unique id for each user | INT(5) | NO | NO | NO | NONE |
|  | **fullname** | full name of user | VARCHAR(50) | NO | NO | NO | NONE |
|  | **company** | name of company that the user works for | VARCHAR(50) | NO | NO | NO | NONE |
|  | **gender** | gender of the user | VARCHAR(20) | YES | NO | NO | NONE |
|  | **age** | age of the user | INT(5) | YES | NO | NO | NONE |
| **flightbooking** | **travelcode** | unique id of each flightbooking | INT(10) | NO | NO | NO | NONE |
|  | **flightfrom** | city where the flight departs from | VARCHAR(50) | NO | NO | NO | NONE |
|  | **flightto** | city where the flight lands | VARCHAR(50) | NO | NO | NO | NONE |
|  | **flighttype** | seat class of the booking (economy, first class, premium) | VARCHAR(50) | NO | NO | NO | NONE |
|  | **flightprice** | price of the flight | DECIMAL(10,2) | NO | NO | NO | NONE |
|  | **flightdate** | date when the flight departs the departure city | VARCHAR(10) | NO | NO | NO | NONE |
|  | **flighttime** | duration of the flight from takeoff to landing | DECIMAL(10,2) | NO | NO | NO | NONE |
|  | **distance** | distance between the departure city and the arrival city | DECIMAL(10,2) | NO | NO | NO | NONE |
|  | **agency** | travel agency that booked the flight for the user | VARCHAR(50) | NO | NO | NO | NONE |
| **hotelreservation** | **hotelname** | name of the hotel | VARCHAR(50) | NO | NO | NO | NONE |
|  | **travelcode** | unique id associated with the booking | INT(10) | NO | NO | NO | NONE |
|  | **location** | city where the hotel is located | VARCHAR(50) | NO | NO | NO | NONE |
|  | **days** | number of nights that the user will be staying at the hotel | INT(5) | NO | NO | NO | NONE |
|  | **hotelprice** | price of the hotel room per night | DECIMAL(10,2) | NO | NO | NO | NONE |
|  | **totalpaid** | total price paid by the user for the duration of their stay at the hotel | DECIMAL(10,2) | NO | NO | NO | NONE |
|  | **checkindate** | date when the user checks in to the hotel | VARCHAR(50) | NO | NO | NO | NONE |

**RELATIONAL SCHEMA**

**users**(**usercode**, name, company, gender, age)

PRIMARY KEY: usercode

**flightbooking(travelcode**, **travelfrom**, travelto, flighttype, flightprice, flightdate, flighttime, distance, agency, *usercode*)

PRIMARY KEY: travelcode, travelfrom

FOREIGN KEY: usercode

**hotelreservation(hotelname, travelcode**, location, days, hotelprice, totalpaid, checkindate, *usercode*)

PRIMARY KEY: hotelname, travelcode

FOREIGN KEY: usercode

**CORRESPONDING TABLES TO STORE THIS DATASET IN MYSQL**

CREATE STATEMENTS

CREATE TABLE users (

usercode INT(5) NOT NULL,

fullname VARCHAR(50) NOT NULL,

company VARCHAR (50) NOT NULL,

gender VARCHAR(20) NOT NULL,

age INT(3) DEFAULT NULL,

PRIMARY KEY (usercode)

);

CREATE TABLE flightbooking (

travelcode INT(10) NOT NULL,

flightfrom VARCHAR(50) NOT NULL,

flightto VARCHAR (50) NOT NULL,

flighttype VARCHAR(50) NOT NULL,

flightprice DECIMAL(10,2) NOT NULL,

flightdate VARCHAR (50) NOT NULL,

flighttime DECIMAL(10,2) NOT NULL,

distance DECIMAL(10,2) NOT NULL,

agency VARCHAR(50) DEFAULT NULL,

usercode INT(5) NOT NULL,

PRIMARY KEY (travelcode, flightfrom),

FOREIGN KEY (usercode) REFERENCES users (usercode)

);

CREATE TABLE hotelreservation (

hotelname VARCHAR(50) NOT NULL,

travelcode INT(10) NOT NULL,

location VARCHAR(50) NOT NULL,

days INT(5) NOT NULL,

hotelprice DECIMAL(10,2) NOT NULL,

totalpaid DECIMAL(10,2) NOT NULL,

checkindate VARCHAR(50) NOT NULL,

usercode INT(5) NOT NULL,

PRIMARY KEY (hotelname, travelcode),

FOREIGN KEY (usercode) REFERENCES users (usercode)

);

**INSERT STATEMENTS**

\*Please see attached SQL file for INSERT statements including VALUES. I have over 140,000 rows, therefore, it would have been too long to insert it here.

INSERT INTO users (

usercode, fullname, company, gender, age)

VALUES (….

);

INSERT INTO flightbooking(

travelcode, travelfrom, traveltoto, flighttype, flightprice, flightdate, distance, agency, usercode)

VALUES (….

);

INSERT INTO hotelreservation(

hotelname, travelcode, location, days, hotelprice, totalpaid, checkindate, usercode)

VALUES (….

);

**PART 3**

***QUERIES***

*I want to start off my data analysis by getting a summary of some descriptive insights about my dataset. All the queries used to obtain this preliminary information is quite simple and similar (i.e. repetitive), however, it is essential when starting such an analytical project, to gain an overview of your data:*

* 1. **Total amount of money spent on business travel over the last 2 years, and the amount of money spent in 2019 vs 2020.**

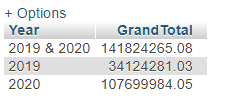
SELECT '2019 & 2020' AS Year, SUM(flightprice) AS GrandTotal FROM flightbooking

UNION

SELECT '2019' AS Year, SUM(flightprice) AS 2019Total FROM flightbooking WHERE flightdate regexp '/2019'

UNION

SELECT '2020' AS Year, sum(flightprice) AS 2020Total FROM flightbooking WHERE flightdate regexp '/2020';



*I got the sum of the flightprice and joined all 3 data sets using the UNION statement – so it comes out in one table. I was also able to create new ‘labels’ on my table which was great!*

*Overall, over ‎R$ 141,824,265.08 (Brazilian Real)**was spent on corporate travel in 2019 and 2020 by the top 5 corporations. There was a huge increase in spending from 2019 (R$ 34,124,281.03) to 2020 (R$ 107,699,984.05) – over 3 time more money was spent. Although this data does not explicitely provide information pertaining to why that occurred, the corporations can use this insights and discuss internally to find out why they spent more money on corporate travel in 2020.*

* 1. **Total amount of money spent on hotel accommodation over the last 2 years, and the amount of money spent in 2019 vs 2020.**

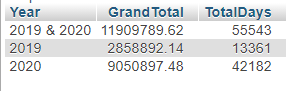
SELECT '2019 & 2020' AS Year, SUM(totalpaid) AS GrandTotal, SUM(days) AS TotalDays FROM hotelreservation

UNION

select '2019'AS Year, SUM(totalpaid) AS 2019Total, SUM(days) AS 2019Days FROM hotelreservation WHERE checkindate REGEXP '/2019'

UNION

SELECT '2020' AS Year, SUM(totalpaid) AS 2020Total, SUM(days) AS 2020Days FROM hotelreservation WHERE checkindate REGEXP '/2020';



*I got the sum of totalpaid and sum of days and joined all 3 data sets using the UNION statement – so it comes out in one table. I used REGEXP because the date column was VARCHAR, not DATE. I was also able to create new ‘labels’ on my table which was great!*

*Overall, over ‎R$ 11909789.62 was spent on hotels in 2019 and 2020 by the top 5 corporations. There was a huge increase in spending from 2019 (R$ 2,858,892.14) to 2020 (R$ 9,050,897.48) – over 3 time more money was spent. Similarly, the number of days spent in hotels more than tripled from 13,361 days to 42,182 days. This data can be used by the corporations to decide if they should be sending their employees on more one-day trips rather than over-night trips, if they want to cut costs.*

* 1. **Total distance travelled by flights and the total time spent travelling over the last 2 years. Additionally, find the breakdown of distance and time travelled by each company over the last 2 years.**

SELECT SUM(distance) AS TotalDistance, SUM(flighttime) TotalTime

FROM flightbooking;

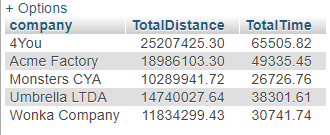
SELECT users.company, SUM(flightbooking.distance) AS TotalDistance, SUM(flightbooking.flighttime) TotalTime

FROM flightbooking

JOIN users

ON users.usercode = flightbooking.usercode

GROUP BY users.company;





*I got the sum of flightdistance and sum of flighttime GROUPED BY company. The data came from 2 different tables so I used the JOIN statement. I could have also just used the WHERE statement, but I wanted to diversify my queries.*

*Overall, 81,057,797.39km was covered by air travel by all 5 corporations. 4You had the highest distance, and Wonka Company had the least. If these companies are seeking to cut down on their carbon footprint, they could use these figures as starting point.*

*Additionally, a total of 210611.38 hours was spent travelling, which is a loss of productive time for employees and the company. These corporations could use this information to figure out how to cut down on travelling so that they can maximize employee time for more productive work.*

* 1. **How many employees travelled on business in 2019 vs in 2020**

SELECT '2019' AS YEAR, COUNT(DISTINCT usercode) AS 2019Total

FROM flightbooking

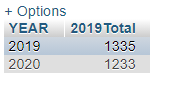
WHERE flightdate REGEXP '/2019'

UNION

SELECT '2020' AS YEAR, COUNT(DISTINCT usercode) AS 2020Total

FROM flightbooking

WHERE flightdate REGEXP '/2020';



*Used the COUNT function to count the DISTINCT usercodes for each year (using REGEXP), and then used UNION to merge the 2 outputs together on one table. Also learned how to create my own labels.*

*Less employees travelled in 2020 than in 2019. However, according to Query #1a, more money was spent on travel in 2020 than in 2019. There could be several reasons for this: flight prices increased; employees flew more firstclass than economy class; etc… This is an interesting insight which can be explored further with market research.*

***\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\****

Now that we have a basic overview of the corporate travel landscape pertaining to our Brazilian companies, let us take a closer look at the details of the data set.

1. **What is the destination that most employees (users) travelled to for business?**

SELECT DISTINCT flightto, COUNT(\*)

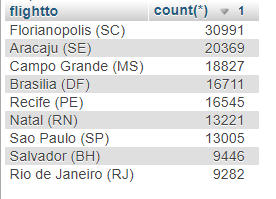
FROM

(SELECT flightto

FROM flightbooking) AS travelcount

GROUP BY flightto

ORDER BY COUNT(\*) DESC



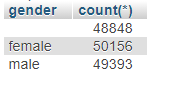
*Didn’t really need the subquery – but basically, I just wanted to count the number of flights going to each flightto destination.*

*This shows that most people flew to the city of Florianopolis (SC) which could mean that it is a business hub. The corporations could use this information when deciding whether to open a new branch in that city. If they have a branch there, it will limit travel and that means they could take advantage of the business opportunities in that city. Furthermore, as a potential business hub, the hotels could use this information as well to decide where to open up new locations.*

1. **What is the gender distribution between men and women? Does one gender group take more business trips than the other?**

SELECT

gender,

 COUNT(\*)

FROM

users,

flightbooking

WHERE

users.usercode = flightbooking.usercode

GROUP BY

gender

*wanted to count the number of male and females who travelled – so I used the COUNT statement, and since the data was coming from 2 different tables, I used the WHERE function to join the tables and then GROUPED by gender*

*Interestingly, more women took more business trips than men. Could this mean that more women hold positions that require business travel and important meetings? We will need to explore further!*

1. **Which hotel received the most reservations and how much did each hotel make in revenue from these reservations?**

SELECT

 hotelname,

location,

COUNT(\*) AS COUNT,

AVG(hotelprice) AS avg\_price,

SUM(totalpaid) AS revenue

FROM

hotelreservation

GROUP BY

hotelname,

location

ORDER BY

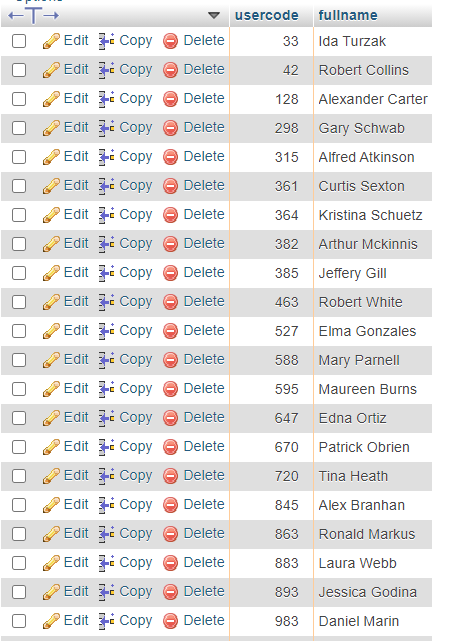
COUNT(\*)

DESC

*So, for this output, I wanted to show 3 key information about the hotel reservations, GROUPED by hotel name. COUNT of how many reservations had been made for each hotel, the average price per night of each hotel, and the total revenue collected (sum of totalpaid).*

*I first grouped by both hotelname and location*

*We can see that although Hotel K had the most reservations (2808), it did not make the most amount of money. The most amount of money was made by Hotel AU (R$ 1,957,377.31). This is because Hotel AU has a higher average price per night. Therefore, Hotel K could use this insight and decide to explore the possibility of increasing their price per night. However, they would need to do more market research on the hotel prices in their location (Salvador (BH)).*

1. **Which users did not stay in hotels during their business travels?**

SELECT

usercode,

fullname

FROM

users

WHERE NOT EXISTS

(

SELECT

usercode

FROM

hotelreservation

WHERE

users.usercode = hotelreservation.usercode

)

*A total of 25 employees did not have hotel bookings.*

*Made a subquery of all the usercodes in hotel reservation using data from both tables (so I had to use the WHERE statement). And then I created an outquery to scan through ALL the people in the database and select the users who were NOT in the subquery – so those who were not associated with hotel reservations. Correlated with my assumption*

1. **Which hotel offers the cheapest nightly rate, and which hotel offers the most expensive nightly rate?**

SELECT DISTINCT 'Max' AS ranking, hotelname, location, hotelprice

FROM hotelreservation

WHERE hotelprice IN (

SELECT MAX(hotelprice)

FROM hotelreservation

)

UNION

SELECT DISTINCT 'Min' AS ranking, hotelname, location, hotelprice

FROM hotelreservation

WHERE hotelprice IN (

SELECT MIN(hotelprice)

FROM hotelreservation

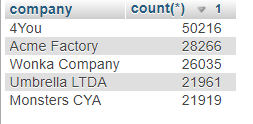
)



*The price difference between the max hotel rate and the minimum one is quite huge (R$ 252.63)*

*First found the MAX hotelprice with the subquery. And then I wanted to find the associated attributes that described that MAX hotel, so I created an outer query to select hotelname, location, hotelprice from the sub squery*

1. ***\*\*THIS IS A THREE-PART QUERY\*\*:***
   1. **Which company has the most business flight activity?**



SELECT company, COUNT(\*)

FROM users, flightbooking

WHERE users.usercode = flightbooking.usercode

GROUP BY company

ORDER BY COUNT(\*) DESC

COUNT number of flights GROUPED by company

*4YOU booked 50,216 flights for its employees in the last 2 years, which is almost twice as many as the other companies.*

* 1. **Which users did the most travel? Which company do these users work for? Did the top traveller come from the top company (see 4a)?**

SELECT

 users.usercode,

users.fullname,

company,

COUNT(\*) AS flight\_count

FROM

users,

flightbooking

WHERE

users.usercode = flightbooking.usercode

GROUP BY

flightbooking.usercode

ORDER BY

flight\_count

DESC

COUNT number of flights taken by each user – so GROUP by usercode – JOIN because of two different tables. The question I want to ask was if the top traveller came from the top company, but since there were many top travellers, I wanted to see if the most amount of top travellers came from the top company, so I had to write another query to count the number of top travellers (so WHERE count = 133), and group by company name. I just used the query in part b as the subquery from part c and grouped by company name, and only kept users whose count was 133 (the max number).

*It looks like there are multiple employees who took 133 flights (which is the top # of flights) – it is necessary to write another query to determine which company has the most top travellers:*

SELECT companys.company, count(companys.company)

FROM (SELECT users.usercode, users.fullname, company, count(\*) as counter

from users

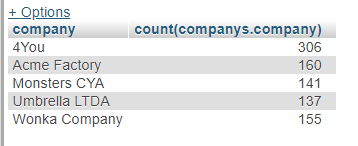
INNER JOIN flightbooking ON users.usercode = flightbooking.usercode

group by flightbooking.usercode

order by count(\*), company) AS companys

WHERE companys.counter = '133'

GROUP BY companys.company



*We can see that most of the top travellers work at 4You. This company should look into getting travel discounts or loyalty points if they have not already – as there could be some substantial savings to be done!*

1. **Let’s take a look at the different classes of flights (economic, first class, premium) that each company purchased.**

SELECT users.company, flightbooking.flighttype, count(flighttype), avg(flightprice), sum(flightprice)

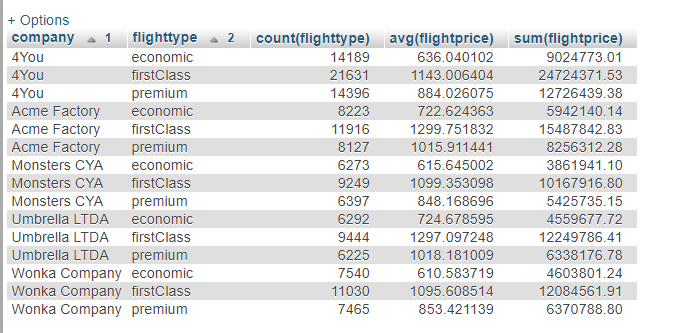
from users, flightbooking

WHERE users.usercode = flightbooking.usercode

GROUP BY users.company, flightbooking.flighttype

ORDER BY company, flighttype

Different statistics about the different flight types. COUNT of number of the different flight types (economy, business, first class), average for each flight type, and sum as well. Grouped by company and by flighttype.



*Several insights can be gleaned from this table. For instance, we see that 4You purchased more first class flights for its employees (21,631) that any other flight type, AND overall, 4You was the company that bought the most first class flights (21,631) out of all the other companies.*

*On average, Umbrella LTDA and Acme Factory spent considerably more on all their flight types compared to the other companies. Therefore, they should perhaps look into finding a better travel agency to work with which gets them cheaper flight deals.*

1. **How many employees are in each age range?**

SELECT

COUNT(case when age >= 0 and age <= 25 then users.usercode end) 0\_25,

COUNT(case when age >= 26 and age <= 35 then users.usercode end) 26\_35,

COUNT(case when age >= 36 and age <= 45 then users.usercode end) 36\_45,

COUNT(case when age >= 46 and age <= 55 then users.usercode end) 46\_55,

COUNT(case when age >= 56 and age <= 65 then users.usercode end) 56\_65,

COUNT(case when age >= 66 then users.usercode end) over\_66

FROM users



I wanted to get an idea of the age distribution of all the employees in the database. So I did some research and learned about the CASE WHEN statement. So basically, this is saying that the system should run through the data and when a number in the age column is between 0-25, they should count the user code, and the label in the output will be 0\_25

*Most people who travel are in the older age groups, 46-55 and 36-45, and the younger employees (0-25) travel the least. This makes sense because usually, more traveling opportunities are given to more senior employees and older people with more experience usually occupy these senior roles. While younger people are more likely to be in entry-level positions which largely require less travel.* ***Assumption:*** *There are no employees over 65, as that is the retirement age in Brazil. My data validates this assumption.*

1. **Average length of stay per age group**

ALTER TABLE users

ADD COLUMN age\_range varchar(10) DEFAULT NULL AFTER age;

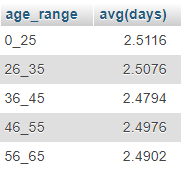
UPDATE users SET age\_range = CASE

when age >= 0 and age <= 25 then '0\_25'

when age >= 26 and age <= 35 then '26\_35'

when age >= 36 and age <= 45 then '36\_45'

when age >= 46 and age <= 55 then '46\_55'

 when age >= 56 and age <= 65 then '56\_65'

when age >= 66 then 'over\_66' END;

SELECT users.age\_range, avg(days)

FROM users, hotelreservation

WHERE users.usercode = hotelreservation.usercode

GROUP BY age\_range;

*There is no correlation between age\_range and average days spent on business travel.*

*To get this information, I chose to create a new column in my users table so that the age ranges could be more permanent and I could use them for more complex queries. So I ALTERed by table to ADD column AFTER the age column – just to keep it orderly*

*Then I UPDATED the new age\_range column using the CASE function. So when between 0 and 25, then the column gets populated with ‘0\_25’. After that step, then I wrote a SELECT query to calculate the average days spend by users grouped by the age range*

1. **In which city are youngest employees most likely to be located?**

SELECT

users.age\_range,

flightbooking.flightfrom,

COUNT(\*)

FROM

users,

flightbooking

WHERE

users.usercode = flightbooking.usercode

GROUP BY

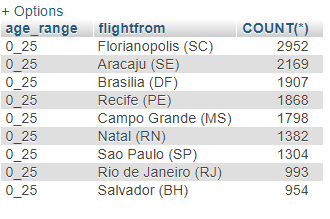
users.age\_range,

flightbooking.flightfrom

HAVING

users.age\_range = '0\_25'

ORDER BY count(\*) DESC



***Assumption: The departure city is where the office is located.***

*0-25 year olds are mostly located in Florianopolis(SC). It is common for younger people to go live in bigger cities in search for opportunities, so companies could centre their recruitment efforts on locations were young people are likely to be located – for entry level jobs.*

Used the GROUP BY and HAVING statements. First select the age range, flightfrom and count of the different flights departing from that destination. Grouped by Age range and then only selected the 0\_25 age range by using the having function. Added an assumption that the departure city is where the home office is located, so that is where the employees live.

1. **On what months are most flights done?**

ALTER TABLE flightbooking

ADD COLUMN month varchar(10) DEFAULT NULL AFTER flightdate;

UPDATE flightbooking SET month = CASE

when flightdate REGEXP '^01/' then 'january'

when flightdate REGEXP '^02/' then 'february'

when flightdate REGEXP '^03/' then 'march'

when flightdate REGEXP '^04/' then 'april'

when flightdate REGEXP '^05/' then 'may'

when flightdate REGEXP '^06/' then 'june'

when flightdate REGEXP '^07/' then 'july'

when flightdate REGEXP '^08/' then 'august'

when flightdate REGEXP '^09/' then 'september'

when flightdate REGEXP '^10/' then 'october'

when flightdate REGEXP '^11/' then 'november'

when flightdate REGEXP '^12/' then 'december'

END;

SELECT month, count(\*)

FROM flightbooking

GROUP BY month

ORDER BY count(\*) DESC;

SELECT month, avg(flightprice)

FROM flightbooking

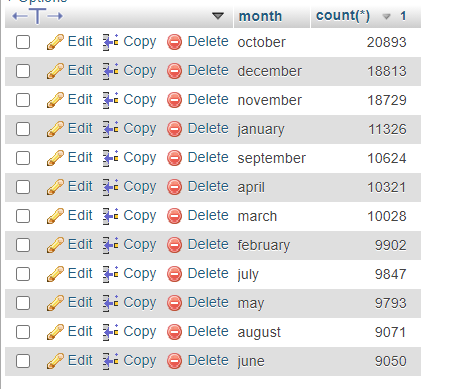
GROUP BY month

ORDER BY avg(flightprice) DESC

ALTERed table to create a new Column called month

Then UPDATED the new column using the CASE function – so whenever a month includes 01, then it means January. I used REGEXP because my dates were VARCHAR not DATE, so that is the best way to do it in my dataset.

Once the new column has been populated, I created 2 SELECT queries to count the number of flightbookings per month, and then the average price of the flight bookings per month. I could have put them in the same table, but I put them in separate tables because I wanted to order them separately to see if the months matched or were correlated



2 - Interestingly, there is no correlation between month and flight price – travel agencies and travel sites should use this information to adjust their prices and make more money



1- Most travel occurs towards the end of the year (October, November, December)

1. **Is there a correlation between distance and days. We are trying to find out if people tend to stay in hotels for more days if they are travelling long distances, to avoid doing 2 long trips back-to-back).**

ALTER TABLE flightbooking

ADD COLUMN avg\_distance VARCHAR(20) DEFAULT NULL AFTER distance;

UPDATE flightbooking SET avg\_distance = CASE

when distance >= 0 and distance <= 200 then '0\_200'

when distance >= 201 and distance <= 400 then '201\_400'

when distance >= 401 and distance <= 600 then '401\_600'

when distance >= 601 and distance <= 800 then '601\_800'

when distance >= 801 and distance <= 1000 then '801\_1000'

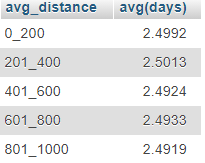
END;

SELECT flightbooking.avg\_distance, avg(days)

FROM flightbooking, hotelreservation

WHERE flightbooking.usercode = hotelreservation.usercode

GROUP BY avg\_distance;



*The output shows that there is no correlation between average distance travelled and average days spent in a hotel, because there is very little variation in the average days spent in hotel.*

ALTERed table to create a new Column called avg\_distance

Then UPDATED the new column using the CASE function – so whenever distance is between 0 to 200 for example, it adds the VARCHAR text ‘0\_200’ in the new avg\_distance column.

Once the new column has been populated, I created a SELECT query to count average days travelled by per avg\_distance range – so GROUP BY average distance.

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

*Let us take a deeper dive into company statistics*

1. **How many employees are there per company?**

SELECT

company,

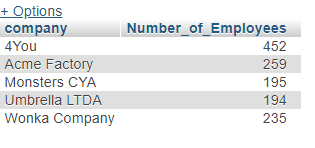
COUNT(usercode) AS Number\_of\_Employees

FROM

users

GROUP BY

Company



*4You has the greatest number of employees in the database, which would explain why they ranked first in a lot of our queries (e.g. amount of money spent on flight, number of flights taken in the last 2 years, etc…)*

*Count number of employees GROUP by company on the user table*

1. **What is the gender distribution in each company (% male and % female)**

SELECT

u1.company,

u2.gender,

COUNT(u2.gender),

(

COUNT(u2.gender) \* 100 / totalemployees.totale

) AS percent

FROM

users u1,

users u2,

(

SELECT

company,

COUNT(gender) AS totale

FROM

users

GROUP BY

company

) AS totalemployees

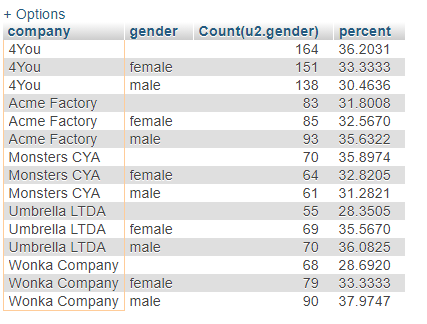
WHERE

u1.usercode = u2.usercode AND u1.company = totalemployees.company

GROUP BY

u1.company,

u2.gender

*Overall, it is great to see that the gender distribution is fairly even between men and women (usually 30% each). 4You and Monsters CYA have slightly more females than males, while the other companies have slightly more males than females.*

This one actually took me quite a while, because I don’t know why but it was hard for me to calculate the percentage. But I eventually figured it out haha! So I used a subquery to get the TOTAL number of male AND female employees in each company – so GROUP by company). Call this subquery totalemployees. Because for the percentage calculation, you need to divide the count of each specific gender by the total of all the people regardless of gender. So then the outerquery used the data derived from the subquery in the percentage calculation. And then overall, the entire query was grouped by company and gender. Since I was getting information from the same table, I used aliases U1 and U2, and then joined them at the end with u1.usercode = u2.usercode

1. **Age distribution at the company**

SELECT

u1.company,

u2.age\_range,

COUNT(u2.gender) AS COUNT,

(

COUNT(u2.age\_range) \* 100 / totalemployees.totale

) AS percent

FROM

users u1,

users u2,

(

SELECT

company,

COUNT(age\_range) AS totale

FROM

users

GROUP BY

company

) AS totalemployees

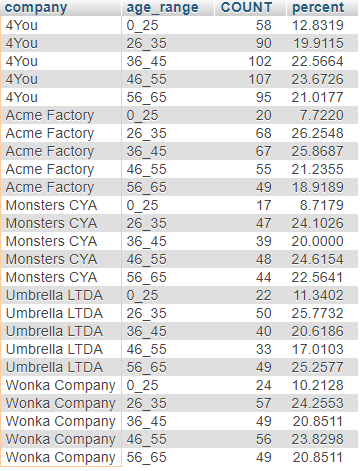
WHERE

u1.usercode = u2.usercode AND u1.company = totalemployees.company

GROUP BY

u1.company,

u2.age\_range



*All companies have more employees in the older age ranges. Acme Factory has the least amount of young people (7.7%) while Umbrella LTDA has the greatest number of oldest employees (56\_65) – 25.3%. There*

This one actually took me quite a while, because I don’t know why but it was hard for me to calculate the percentage. But I eventually figured it out haha! So I used a subquery to get the TOTAL number of employees in each company – so GROUP by company). Call this subquery totalemployees. Because for the percentage calculation, you need to divide the count of each specific age\_ranges by the total of all the people regardless of age\_range. So then the outerquery used the data derived from the subquery in the percentage calculation. And then overall, the entire query was grouped by company and age\_range. Since I was getting information from the same table, I used aliases U1 and U2, and then joined them at the end with u1.usercode = u2.usercode

1. **Let’s have a breakdown of the Total amount of money that each company spent on flights and on hotels separately, as well as the Grand Sum they spent on both flights and hotels over the last 2 years.**

SELECT

users.company,

SUM(flightbooking.flightprice) AS TotalFlights,

SUM(hotelreservation.totalpaid) AS TotalHotel,

(

SUM(hotelreservation.totalpaid) + SUM(flightbooking.flightprice)

) AS TotalSpend

FROM

users,

flightbooking,

hotelreservation

WHERE

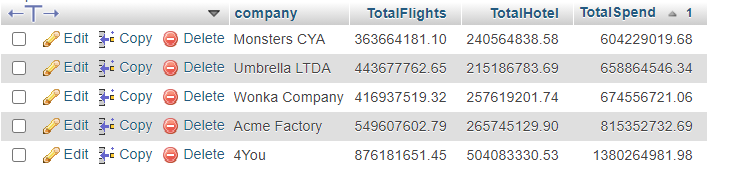
users.usercode = flightbooking.usercode AND users.usercode = hotelreservation.usercode

GROUP BY

users.company

ORDER BY

TotalSpend



*As expected based on other queries so far, 4You scores the highest in all Total figures. They spent almost double the amount of the other companies.*

*Got the sum of flightprice AS TOTALprice and sum of hotelprice AS TotalHotel – then the sum of both TOTAlpaid and hotelprice as TOTALspend – and this was from 3 tables (users, filghtbooking and hotelreservation), so I used the WHERE statement to JOIN them on the usercode attribute, and then grouped by company name and ordered by total spend.*

1. **SCENARIO: Since October, November and December are usually the busiest travel month (as determined by Query #12), Hotel AU wants to temporarily VIEW all of its upcoming reservations in those months to ensure that they are ready for the volume of people checking in.**

CREATE VIEW hotelau\_oct\_nov\_dec AS SELECT

hotelreservation.usercode,

users.fullname,

hotelreservation.checkindate,

hotelreservation.days

FROM

users

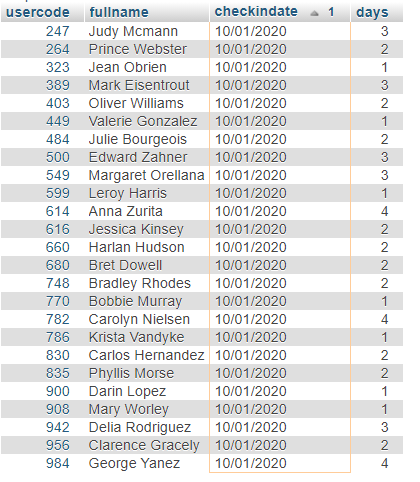
JOIN hotelreservation ON users.usercode = hotelreservation.usercode

WHERE

hotelreservation.hotelname = 'Hotel AU' AND hotelreservation.checkindate NOT REGEXP '/2019' AND hotelreservation.checkindate REGEXP '^(10/|11/|12)'

ORDER BY

hotelreservation.checkindate



*There are a total of 448 upcoming reservations! Hotel AU will be very busy!*

Used the VIEW statement to create a temporary table. SELECT information, from 2 different tables, used the JOIN statement ON usercode, and then I only wanted Hotel AU, and did not want 2019, and also only wanted October, November and December – so I used REGEXP

1. **Hotel AU knows from historical experience that business travellers who only book for 1 night are more likely to cancel. They would like to calculate how much minimum revenue they can expect from their busy months (Oct, Nov, Dec) even if the guests who booked for 1 night cancel.**

SELECT

SUM(

subset.days \* subset.hotelprice

) AS minimin\_revenue

FROM

(

SELECT

hotelreservation.hotelprice,

hotelreservation.usercode,

users.fullname,

hotelreservation.checkindate,

hotelreservation.days

FROM

users

JOIN hotelreservation ON users.usercode = hotelreservation.usercode

WHERE

hotelreservation.hotelname = 'Hotel AU' AND hotelreservation.checkindate NOT REGEXP '/2019' AND hotelreservation.checkindate REGEXP '^(10/|11/|12)' AND hotelreservation.days > 1

) AS subset

Used the query from Q18 as a subquery, and then calculated the sumproduct of days times hotelprice to get the minimum revenue.

*Even if all the one-night reservations are cancelled, Hotel AU can still expect to make a minimum of R$ 304,070.76 in the peak season.*

***Assumption****: None of the reservations that are more than one night cancel.*

1. **SCENARIO: There has been a closure of the airport in Natal (RN). Travel agency ‘FlyingDrops’ wants to see all the upcoming flights that it has booked where employees are either departing from or landing at the airport in Natal (RN). With this information, they will be able to rebook their flights via another airport.**

SELECT

flightbooking.usercode,

users.fullname,

flightbooking.flighttype,

flightbooking.flightdate

FROM

users

JOIN flightbooking ON users.usercode = flightbooking.usercode

WHERE

(

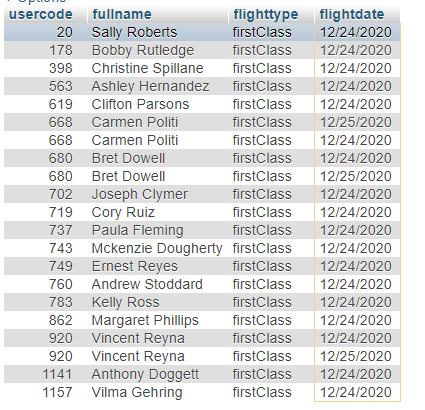
flightbooking.flightto = 'Natal (RN)' OR flightbooking.flightfrom = 'Natal (RN)'

) AND flightbooking.agency = 'FlyingDrops' AND(

flightbooking.flightdate = '12/24/2020' OR flightbooking.flightdate = '12/25/2020'

)

**Wanted to get** information from 2 different rables, and only wanted data that included city Natal(RN), agency ‘FlyingDrops’, and dates December 24 and 25 so I used WHERE statement and REGEXP



*FlyingDrops will have to rebook a total of 21 flights. Interestingly, they are all first class flights.*